EXCERPT FROM THE
PROCEEDINGS

OF THE

SEVENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM WEDNESDAY SESSIONS VOLUME I

Acquisition Research
Creating Synergy for Informed Change

May 12 - 13, 2010

Published: 30 April 2010

			Form Approved OMB No. 0704-0188

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **MAY 2010**	2. REPORT TYPE	3. DATES COVERED **00-00-2010 to 00-00-2010**
4. TITLE AND SUBTITLE **Improved Software Testing for Open Architecture**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Postgraduate School,Computer Science Department,Monterey,CA,93943**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
7th Annual Acquisition Research Symposium to be held May 12-13, 2010 in Monterey, California

14. ABSTRACT
Applying traditional manual US Navy testing practices to OA systems will limit many benefits of OA, such as system scalability, rapid configuration changes, and effective component reuse. Pairing profile-driven automated software testing with test reduction techniques should enable these benefits and keep resource requirements at feasible levels. Test cases generated by operational profiles have been shown to be more effective than those developed by other methods, such as random or selective testing, and more resourceefficient than exhaustive approaches. This research effort increases the fidelity of the operational profile, creating an environment model referred to as a High-Fidelity Profile Model (HFPM) that can statistically describe individual software inputs. Samples from the HFPM?s probability distributions can generate operationally realistic or overly-stressful test cases for software modules under test. This process can be automated and paired with output checking functions, enabling automated effective software testing, and potentially improving reliability. Such models would be ideal for US Navy Open Architecture (OA) software because of the defined interface standards. HFPMs can enable effective testing in software reuse applications and are ideal for testing multiple releases of maturing software. This research defines the HFPM, presents a methodology to develop, validate, and apply it.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**38**	

The research presented at the symposium was supported by the Acquisition Chair of the Graduate School of Business & Public Policy at the Naval Postgraduate School.

To request Defense Acquisition Research or to become a research sponsor, please contact:

NPS Acquisition Research Program
Attn: James B. Greene, RADM, USN, (Ret.)
Acquisition Chair
Graduate School of Business and Public Policy
Naval Postgraduate School
555 Dyer Road, Room 332
Monterey, CA 93943-5103
Tel: (831) 656-2092
Fax: (831) 656-2253
E-mail: jbgreene@nps.edu

Copies of the Acquisition Sponsored Research Reports may be printed from our website www.acquisitionresearch.net

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY   **2**
NAVAL POSTGRADUATE SCHOOL

Improved Software Testing for Open Architecture

Valdis Berzins—Valdis Berzins is a Professor of Computer Science at the Naval Postgraduate School. His research interests include software engineering, software architecture, computer-aided design, and software evolution. His work includes software testing, reuse, automatic software generation, architecture, requirements, prototyping, re-engineering, specification languages, and engineering databases. Berzins received BS, MS, EE, and PhD degrees from MIT and has been on the faculty at the University of Texas and the University of Minnesota. He has developed several specification languages, software tools for computer-aided software design, and fundamental theory of software merging.

Valdis Berzins
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943
Phone: 831-656-2610
E-mail: berzins@nps.edu

Paul Dailey—Paul Dailey is a systems engineer with the Naval Postgraduate School and has worked for the Department of the Navy for seven years, including as a test and evaluation engineer for the Naval Surface Warfare Center, Port Hueneme Division, Detachment Louisville from 2002 to 2009. He holds a MS in Systems Engineering from the Naval Postgraduate School and a BS in Electrical Engineering from the University of Louisville. He is currently pursuing a PhD in Software Engineering from the Naval Postgraduate School, focusing his research on the automated testing of software.

Paul Dailey
Software Engineering PhD Student
Naval Postgraduate School
Monterey, CA 93943
E-mail: prdailey@nps.edu

Abstract

Applying traditional manual US Navy testing practices to OA systems will limit many benefits of OA, such as system scalability, rapid configuration changes, and effective component reuse. Pairing profile-driven automated software testing with test reduction techniques should enable these benefits and keep resource requirements at feasible levels. Test cases generated by operational profiles have been shown to be more effective than those developed by other methods, such as random or selective testing, and more resource-efficient than exhaustive approaches. This research effort increases the fidelity of the operational profile, creating an environment model referred to as a High-Fidelity Profile Model (HFPM) that can statistically describe individual software inputs. Samples from the HFPM's probability distributions can generate operationally realistic or overly-stressful test cases for software modules under test. This process can be automated and paired with output checking functions, enabling automated effective software testing, and potentially improving reliability. Such models would be ideal for US Navy Open Architecture (OA) software because of the defined interface standards. HFPMs can enable effective testing in software reuse applications and are ideal for testing multiple releases of maturing software. This research defines the HFPM, presents a methodology to develop, validate, and apply it.

Keywords: Software Testing, Software Reliability, Operational Profile, Software Reuse, Open Architecture

Introduction

Current software testing methods will limit some of the key benefits that Open Architecture (OA) can provide for the US Navy. More specifically, the ability to rapidly change a system's configuration in order to meet new requirements is possible when using an OA but if current Test and Evaluation (T&E) practices and policies are applied, the updated system will likely not be fielded in a timely manner. With the ability to rapidly update software comes a need to rapidly field that software (Berzins & Dailey, 2009).

In order to rapidly field US Navy combat and weapons system software, two new approaches are required. First, the current software testing process needs to be changed from a manually conducted process to an automated process that provides better test coverage for a given cost and period of time. Second, the total amount of testing required should be safely reduced to a minima acceptable level. Instead of conducting complete end-to-end testing after every configuration change, testing should only be conducted where necessary. The ability to test more rapidly while providing better coverage combined with the ability to determine when retesting is not necessary should enable the ability to rapidly field OA combat and weapon system software (Berzins & Dailey, 2009).

Model Driven Automated Software Testing

The recommended automated software testing process, outlined in detail by Dailey, Berzins, and Luqi (2009; 2010), focuses on developing a High-Fidelity Profile Model (HFPM) for each software component under test (SUT) and then using it to automatically generate test cases, execute test cases, check SUT outputs and analyze the results. Analyzing the results automatically can be challenging for services with new or modified requirements, but can be accomplished easily and economically for components whose behavior is not supposed to change from the previous release. This can be done by running both the new and the previous version of the software component on each input generated by the HFPM and then comparing the results. That process is easy to automate.

The HFPM contains High-Fidelity Profiles (HFPs), which are validated probability distribution functions (PDFs) that characterize the component's environment. Operationally-realistic or stress-inducing test cases are automatically created by sampling from those HFPs and processing the samples through test case generation algorithms. Once generated, the test cases are queued up for automated SUT execution by the software tools implementing the HFPM. Following execution, output analysis algorithms integrated into the HFPM, are used to automatically check the test case outputs and calculate the resulting reliability of the SUT with respect to the HFPs used in testing. The overall process (Figure 1) and the HFPM functional concept (Figure 2) are outlined below. For a more detailed description, see Dailey and Luqi (2010).
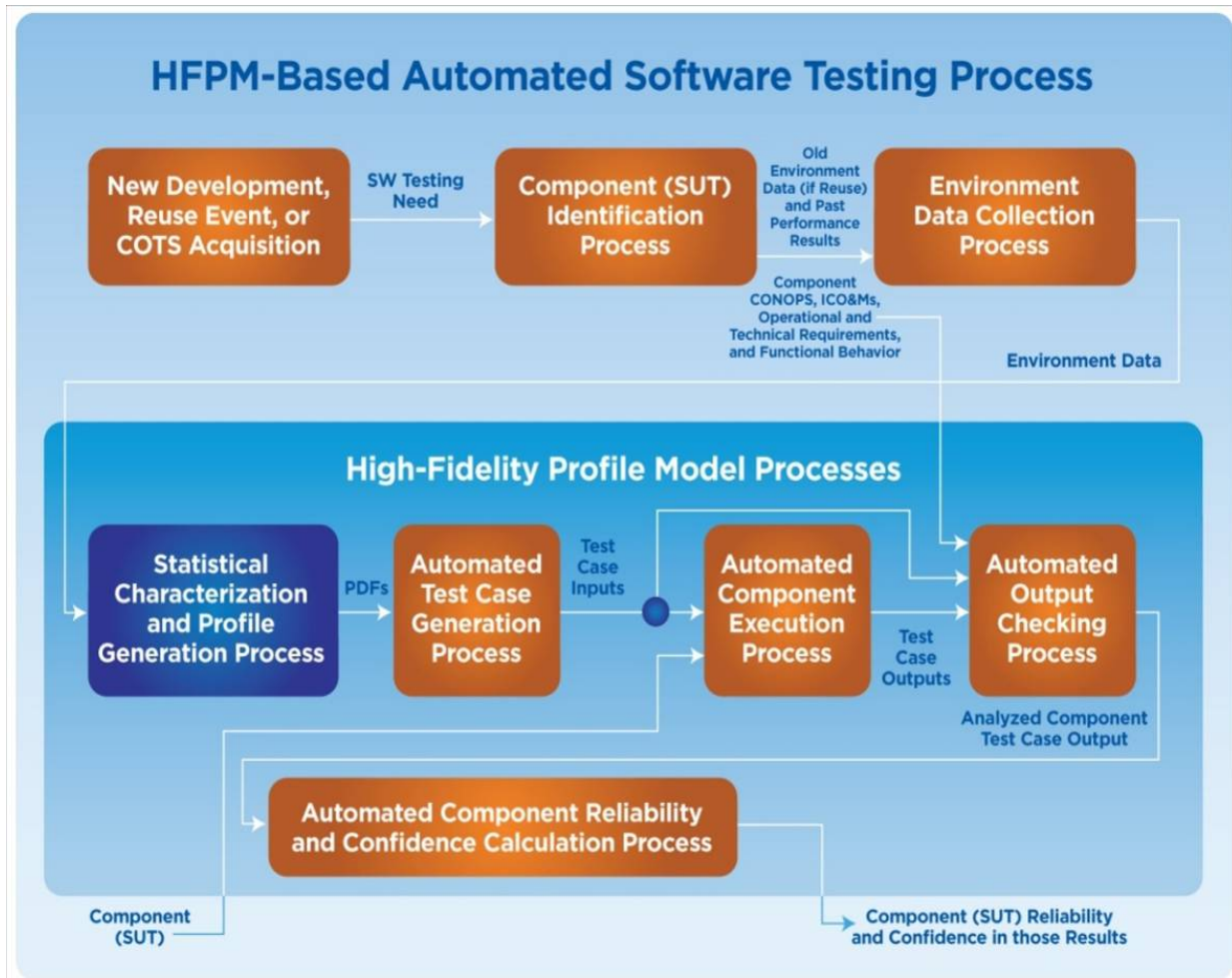
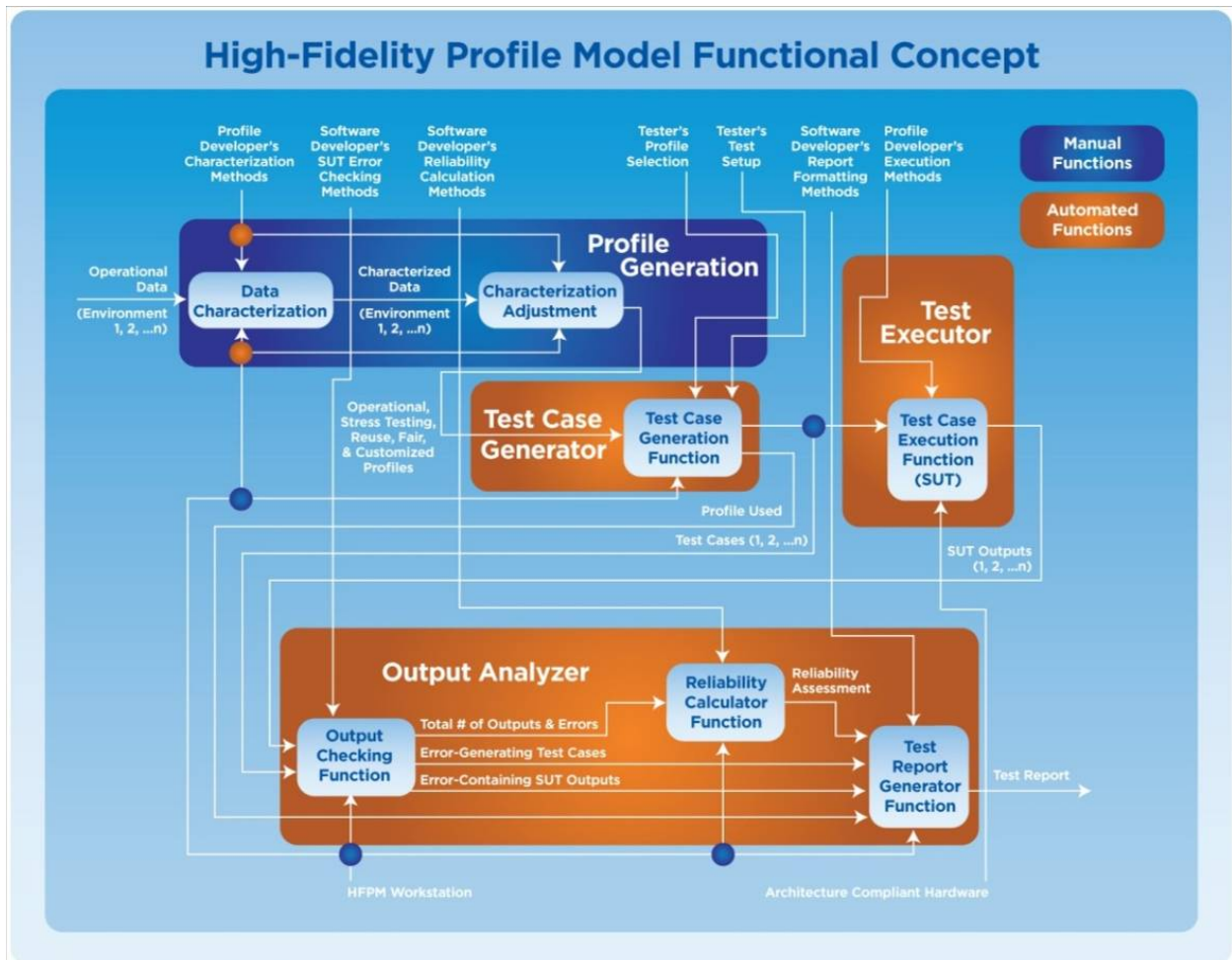**Figure 1. HFPM-Based Automated Testing Process**
(Dailey & Luqi, 2010)

Figure 2. HFPM Functional Concept
(Dailey & Luqi, 2010)

Application to US Navy Acquisition

In order to make the process described above work for US Navy acquisition, it should be employed in a way that enables the HFPM model to be used by all relevant commands that play a role in software development or T&E. This type of focus provides a common practice across the acquisition testing community with the ability for customization for specific roles. The HFPM should be developed in parallel with new components and should be created for a component when acquired off the commercial shelf or in reuse applications where one does not yet exist. The research, development and acquisition agency should use the HFPM to check each component as it is developed and/or integrated into its specific operating environment until such time that the component is ready for Independent Validation and Verification (IV&V). At that time, the component along with the HFPM, are passed to an IV&V test team, which has the ability to modify the HFPs as desired, for Developmental Testing (DT). The IV&V test team can be another group of independent testers in the same command as the software developers or they can be part of the In-Service Engineering Agency (ISEA) responsible for maintaining the software once fielded. This level of DT is generally the most stressful type of testing, focused on

identifying bugs by wider ranges of test inputs than expected in the nominal operating environment.

Once the DT IV&V testing is complete, the results along with the profile(s) used in the testing are passed back to the software development team. If bugs exist that require correction, the software development team can make the proper changes, update the configuration, test internally and send out for another round of DT IV&V. If the software has reached a desired level of maturity for field use, the software component is sent out for Operational Test (OT) certification. OT should be conducted by a command outside of the software development and ISEA, such as the Commander Operational Test and Evaluation Force (COMOPTEVFOR), ensuring independent certification and utilizing more operationally realistic HFPs for test case generation. Often however, such OT agencies do not have the technical expertise to evaluate all types of software. In such cases, members of the software development team can become OT trusted agents and provide support for OT evaluation under control and supervision of the primary OT command. If OT is unsuccessful, the test results and profile(s) used are sent back to the software development team for analysis and correction. Upon successful completion of OT, results are passed back to the software development team and the software is certified for deployment. This concept is illustrated in Figure 3.
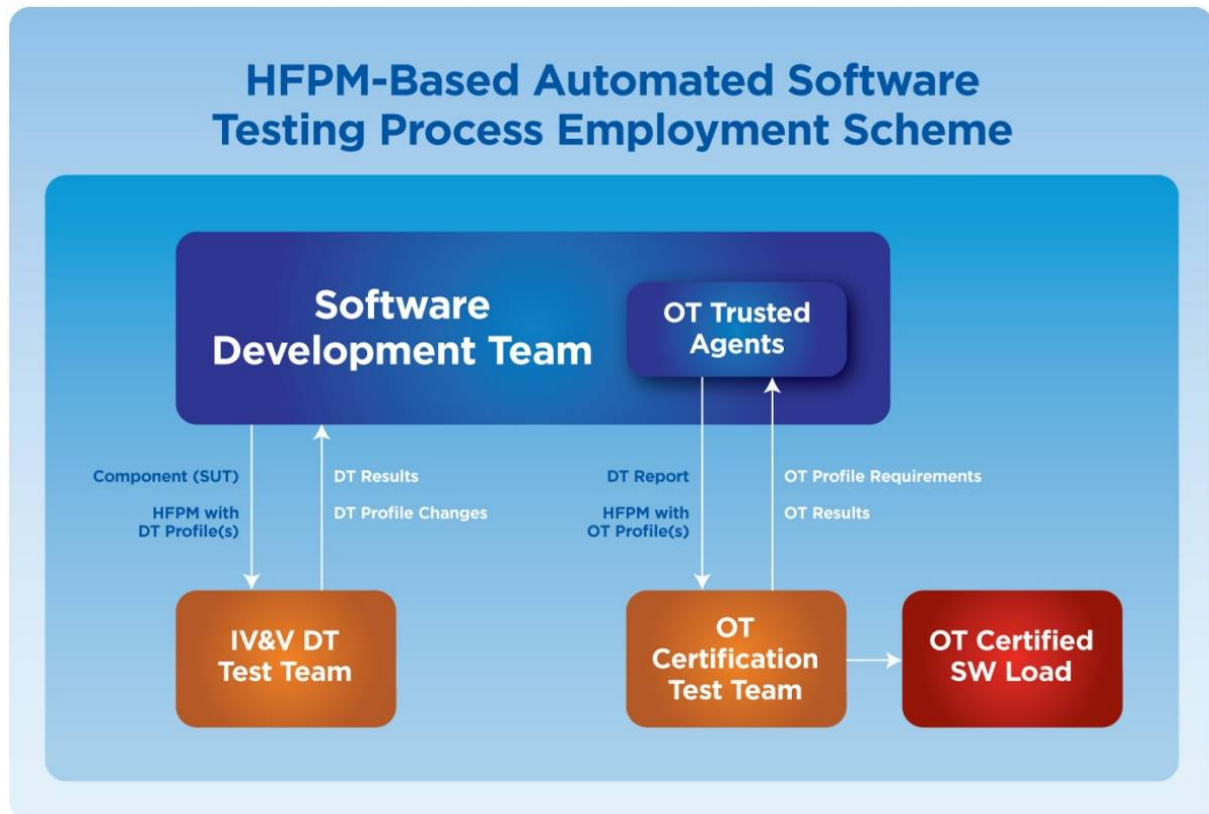

Figure 3.   HFPM-Based Automated Software Testing Process Employment Scheme
(Dailey & Luqi, 2010)

Deriving HFPs from Historical Data

The most important element of the HFPM-driven automated testing process is deriving the HFP(s) for use in automated test case generation as the reliability calculated during testing is only accurate relative to the HFP(s) used. If we develop HFPs characterizing several different deployment environments, the methods described in this paper can be used to determine the reliabilities to be expected in each deployment environment. These can vary considerably.

Collecting Historical Data

The first step in deriving HFPs from historical data is collecting the historical data. To effectively do this, the component to be tested must be understood, including its operational and technical requirements, functional behavior, and expected inputs and outputs. Once all the component inputs and outputs are identified and defined operationally and functionally, the next task is to collect data that can directly or indirectly be used to form characterizations of the expected component inputs in the operating environment.

Depending on the specific application and information available, any type of historical or environment data can potentially be useful in this process. The most ideal case is to obtain actual input data that will be processed by the component in the new environment and directly characterize that data. If this is not obtainable, other indirect but relevant data can be collected and characterized along with information that relates the collected data to the SUT inputs. For applications where the operating environment is not known, a method proposed by Voas (2000) can be helpful if access to the end users during development is possible. In this process, an instrumentation tool is used to collect data from fielded software that can then be used to generate accurate operational profiles. If access to the end users is not possible, it is up to the software development and acquisition team to determine how to best collect useful environment data in each specific application for analysis and HFP generation. Specific methods could include trial data collection efforts during training exercises, Advanced Concept Technology Demonstrations (ACTDs), modeling and simulation, or technical intelligence collection and analysis. Once collected and characterized, indirect data may require further processing by input test case generation algorithms if necessary, in order to transform samples from those characterizations into usable test case inputs.

Characterizing Historical Data

Once a particular set of raw environment data is collected and related to the specific component input(s), the data can be analyzed using one of many established data characterization methods and available commercial tools for HFP PDF generation. One such example is the Matlab® Dfittool application within the Statistics Toolbox® ("Dfittool," 2009). Regardless of the tool used, parametric methods such as Maximum Likelihood Parameter Estimation, and Maximum *A Posteriori* Probability Estimation, or non-parametric methods such as the Histogram, Kernel Density Estimation (KDE) (Wikipedia et al., 2009), or Parzen Neural Network (PNN) (Trentin, 2006) methods can be applied to generate HFP PDFs using available raw environment data. Parametric methods should be used when an understanding of the data is available prior to characterization. If there is no such prior understanding of the data, nonparametric methods can be used more effectively. The desired tool(s) used to perform the necessary analysis should have the flexibility to modify

the method used for calculation in order to compare methods and determine the best type of PDF fit. The output of this analysis process should be one or more PDFs that can be used to either directly or indirectly generate test case inputs based on samples from those PDFs. Direct examples include applications where a sample from the PDF can be used as a component input. Indirect examples include PFD samples that require further processing in order to generate component inputs. These PDFs are referred to as HFPs in this study.

Simple Example of Deriving HFPs from Historical Data

Dailey (2010) illustrated the concept creating HFPs from collected environment data. In the example, performance data on various small boat platforms from a US Navy study was acquired and modeled using Matlab®. The US Navy data provided the following data on six different types of small boat platforms:

Table 1.   Small Boat Collected Data
(Dailey, 2010)

Platform	Max Velocity (kts)	Boat Length (m)	Acceleration (kts/sec)	Deceleration (kts/sec)	Turning Rates (deg/sec)	Speed Loss in Turns (deg/sec)
Boghammer	40	13	1.5	4	10	10
FB 38	50	11.85	2.5	4.3	15	12
7m RHIB	27	7.25	2.5	4.2	28	7
Boston Whaler	36	6.78	2.5	4	30	8
Zodiac	23	4.7	6.25	4.5	32	5
Wave Runner	44	3.66	6.25	4.4	47	15

The data in Table 1 was entered into Matlab® and then characterized using the Statistics Toolbox® *dfittool* resulting in a HFP PDF and inverse cumulative distribution function (Inverse CDF) for each of the parameters. Due to the limited number of points per parameter and the lack of specific knowledge on the specific type of distribution applicable to each parameter, the nonparametric KDE calculation was used to characterize the data. The KDE function is:

$$\hat{f}(x) - \frac{1}{n}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

where *K* is some kernel and h is a smoothing parameter called the bandwidth ("Kernel Density," n.d.). In this case, *K* was taken to be a standard Gaussian function.

Several iterations of characterizations were generated taking into account the actual data as well as establishing logical finite ranges for each parameter. The result is a collection of distributions that effectively describes a notional small boat platform from a technical perspective. Two HFPs generated from this data can be seen below.
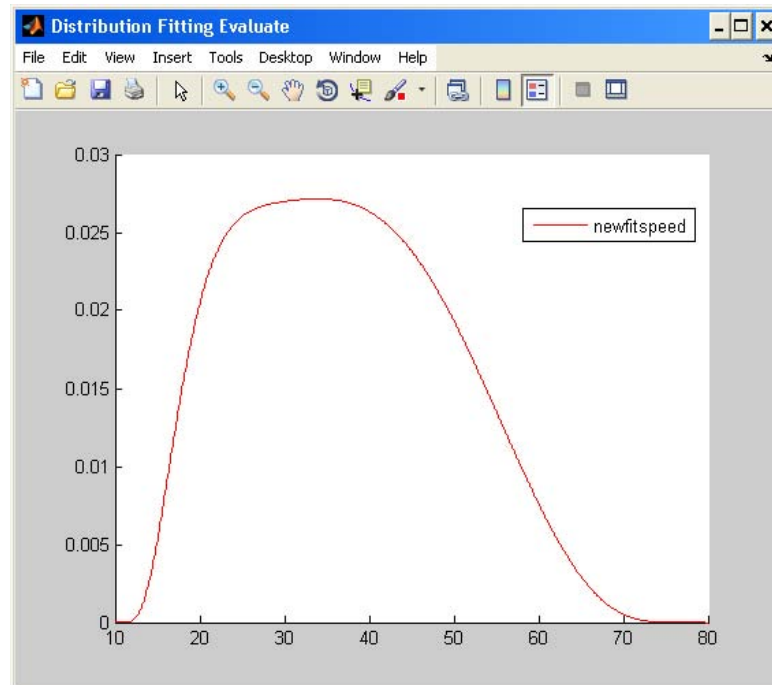
Figure 4.   Notional Small Boat Maximum Velocity PDF (Knots)
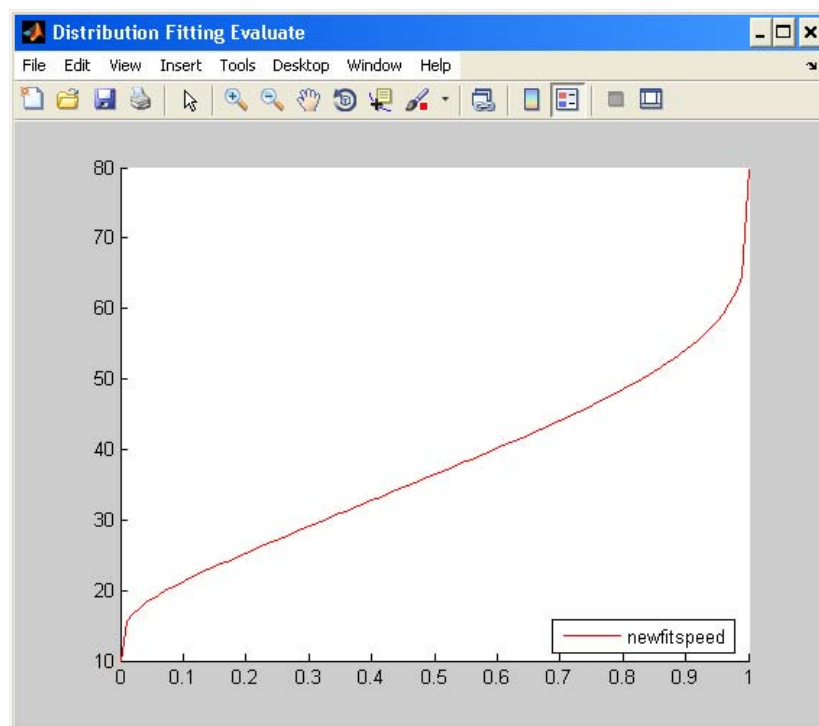(Dailey, 2010)

Figure 5.   Notional Small Boat Maximum Velocity Inverse CDF (Knots)
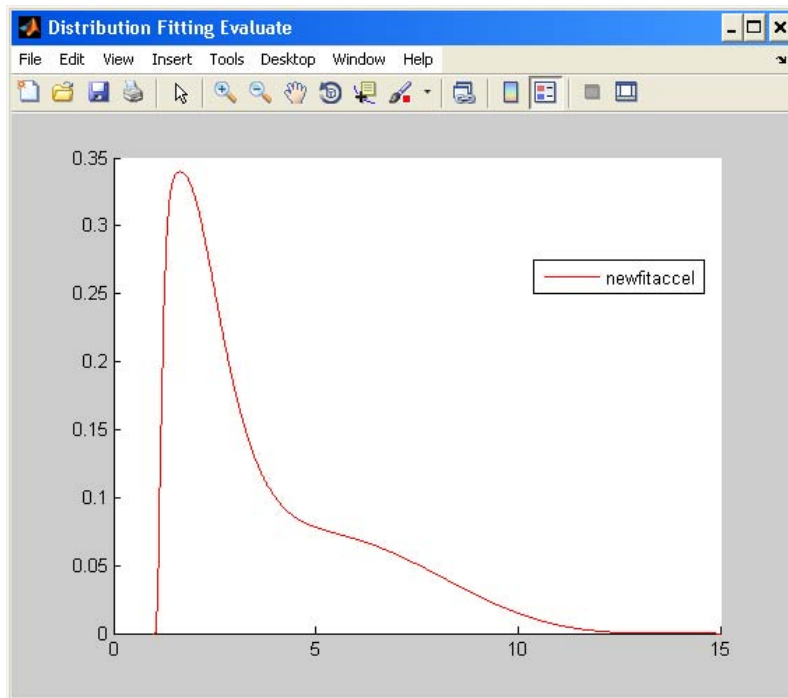(Dailey, 2010)

Figure 6.   Notional Small Boat Acceleration PDF (Knots/Second)
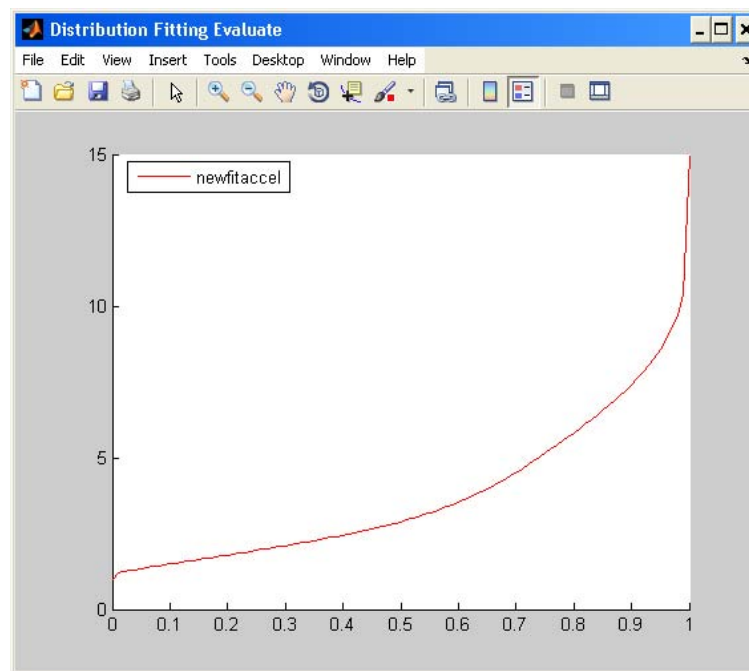(Dailey, 2010)

Figure 7.   Notional Small Boat Acceleration Inverse CDF (Knots/Second)
(Dailey, 2010)

The HFP functions generated above were exported to the Matlab® workspace for use in automated test case generation as part of a HFPM concept demonstration prototype. For more information, see Dailey (2010).

Validating High-Fidelity Profiles

Since testing results from this process are only valid with respect to the HFP(s) used to generate the test cases, it is important to take measures to check their validity. In any application, a qualitative analysis of the HFP(s) should be conducted by subject matter experts to ensure that the derived profile(s) provide adequate coverage for testing. In addition to the qualitative assessment, it would be very useful to define a quantitative process to perform this function. When trying to determine the best characterization method, it is possible to compare the different methods taking into account the methods themselves as well as their results. Various methods currently under investigation to assess the best characterization method include the use of Bayesian Information Criterion (BIC) and goodness of fit tests.

BIC can be used to compare multiple alternative parametric models with different numbers of parameters of a particular environment. When estimating parameters using maximum likelihood estimation, it is possible to modify or increase the likelihood using additional parameters, but this also can result in overfitting. In this method, the model with the lowest BIC score has the best fit. This technique does not apply to non-parametric characterizations such as KDE, but is useful for deciding between different parametric techniques. In addition to BIC, other similar approaches, such as the Akaike Information Criterion (AIC), also exist. BIC applies a stronger penalty than AIC for having additional parameters. The formula for the BIC is as follows:

$$(-2)\ln\ p(x|k) \approx BIC = (-2)\ln(L) + (k)\ln(n)$$

where x is the observed data; n is the number of data points in x; k is the number of free parameters to be estimated; and L is the maximized value of the likelihood function for the estimated model ("Bayesian Information," n.d.).

Another approach for comparing different characterization methods is to perform a goodness of fit test for each characterization to the actual empirical data. One specific type of calculating the goodness of fit of a PDF to an empirical distribution is the Cramér-von-Mises criterion. It is defined as:

$$\omega^2 = \int_{-\infty}^{\infty} [F_n(x) - F^*(x)]^2 dF^*(x)$$

where $F^*(x)$ is the characterized distribution and $F_n(x)$ is the empirical environment data distribution ("Cramér-von-Mises," n.d.).

The methods described above are useful for comparing different HFPs to determine the best method. Ongoing research is being conducted to determine the level of confidence in the HFP with respect to the sample size of empirical environment data. This would be beneficial as it can be used to determine how much environmental data collection is adequate.

Deriving Stress-Testing HFPs from Historical Models

By definition, stress testing exercises a software system beyond the range of normal operating conditions. There are two basic approaches to this–black box and clear box. Black box approaches can be combined with the profile model transformations described in this

section to carry out automated stress testing that can support statistical reliability estimates relative to the stress testing profile(s) (PDF(s)). The black box approaches described in sections 6.1-6.3 can be combined with the method for reducing retesting of reusable components described in Berzins and Dailey (2009) to eliminate redundant repetition of test cases from the previously tested ranges.

Clear box approaches are heuristic methods that seek to uncover particular types of errors. Although clear box criteria can be applied using stress-profiles, other methods should also be considered, as discussed in more detail in sections 6.4 and 6.5.

Standard Deviation Based Methods

The simplest kind of stress testing profile is based on the mean and standard deviation of the HFPM that characterizes the expected operating conditions (Berzins & Dailey, 2009). This approach is applicable to numerical data types and uses a distribution that exercises two intervals symmetrically placed about the mean, from one to N standard deviations set off from the mean in both directions. The parameter N determines how far beyond the expected operating range will be exercised by the stress test. We recommend a series of stress tests with increasing values of N such as (10, 100, 1000, …) up to the entire range supported by the underlying data type.

The approach can readily be generalized to vector data types by choosing a uniform distribution that takes the form of a ring (in 2 dimensions) or a shell (in 3 or more dimensions). The distribution is centered on the mean of the HFPM, and the radius from the center ranges from 1 to N standard deviations. If the HFPM is not isotropic (not the same in all directions), an ellipsoid with different radii along each axis can be used, derived from the covariance matrix of an HFPM over a 2 or more dimensional input space.

Scale Expanding Transformations

Another approach that works for numerical or vector valued inputs is to use a scale expanding transformation. If the HFPM is a distribution $P(x-m)$ where m is the mean of the HFPM, then the stress testing profile derived via the approach in $P((x-m)/s),$ where s is a numerical scale factor. The stress testing profile then has the same mean as the HFPM, and a standard deviation that is s times larger. The shape and orientation of the stress profile are similar to the original, but spread out more by a factor of s, which is similar to the parameter N in the previous section. We recommend a sequence of tests with $s = [10, 100, 1000, …]$ for applying this method.

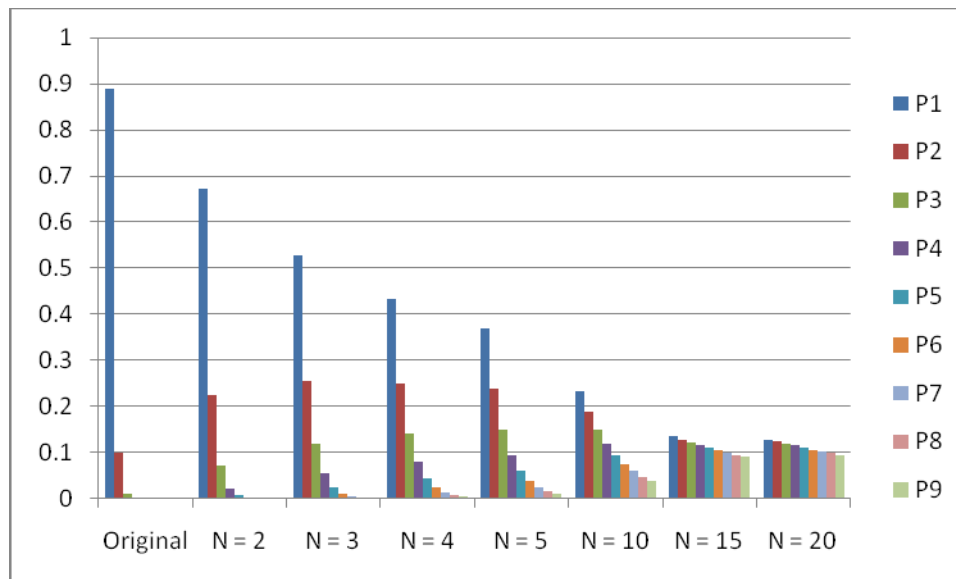## Probability Scaling Transformations

The approaches described in sections 6.1 and 6.2 apply only to numerical or vector-valued input data. In contrast, probability scaling transformations apply to any kind of input data, including discrete enumerations such as classification categories and other non-numerical data types. For a HFPM with a distribution $P(x)$ the stress testing profile derived using this approach is proportional to $P(x)^{1/N}$, where N is a numerical parameter with $N > 1$, and where the proportionally constant must be chosen to normalize the distribution to make all probabilities add up to 1. This family of transformations increases the probabilities of rare events and decreases the probabilities of the frequent ones, as illustrated by the example shown in Table 2.

Table 1.   Original and Derived Probabilities

	Original	N = 2	N = 3	N = 4	N = 5	N = 10	N = 15	N = 20
P1	0.88888889	0.670925	0.526601	0.432891	0.369481	0.233181	0.134859	0.128692
P2	0.1	0.225035	0.254214	0.250707	0.238684	0.187417	0.128468	0.12409
P3	0.01	0.071162	0.117996	0.140983	0.150599	0.14887	0.12206	0.119418
P4	0.001	0.022504	0.054769	0.079281	0.095022	0.118252	0.115971	0.114922
P5	0.0001	0.007116	0.025421	0.044583	0.059955	0.093931	0.110186	0.110595
P6	0.00001	0.00225	0.0118	0.025071	0.037829	0.074612	0.10469	0.106432
P7	0.000001	0.000712	0.005477	0.014098	0.023868	0.059266	0.099468	0.102425
P8	0.0000001	0.000225	0.002542	0.007928	0.01506	0.047077	0.094506	0.098568
P9	0.00000001	7.12E-05	0.00118	0.004458	0.009502	0.037395	0.089792	0.094857

Table 2 shows an original PDF and a series of transformed and renormalized derived stress testing PDFs. Note that the probabilities in each column add up to 1 and that the original distribution spans a wide range of frequencies of occurrence. These distributions are shown as bar graphs in Figure 8.

Figure 8.   Original and Derived Probabilities

The transformations increase the proportions of the rare cases in the stress testing samples, while preserving the rank ordering of the probabilities. The degree of enhancement of the rare events increases with the parameter *N*.

Dominance Relations and Stress Testing

Stress testing does not have to be done solely using HFPM's. Another useful approach is based on the concept of dominance. One test case *dominates* another one if the first one will expose at least as many software faults as the second, and may expose more. Even if there is not a single test case that dominates all of the others, often there will be some that are more likely to expose errors than others. This approach is particularly useful when the tester is focusing on a specific class of errors. Many of the commonly used

testing heuristics are based on this idea and can contribute to efficient testing. A representative sample of these is listed in Table 3, organized by the error type addressed by each.

Table 2.   Focused Stress Testing Strategies

Error Type	Heuristics for choosing stress test cases
Numeric Overflow	Largest and smallest representable numbers
Buffer Overflow	Very long input string
Free Storage Overflow	Create many new objects
Wrong Conditional Logic	Data values close to the both sides of an interval boundary
Unprotected Pointers	Null pointer
Unprotected Division	Zero

Coverage Criteria and Stress Testing

Traditional coverage criteria, such as statement coverage and branch coverage, are useful for checking low probability paths through the software. This can be an important defense against unwanted features deliberately placed in the code by malicious insiders. For example, an "Easter Egg" is a hidden feature function in the software that is triggered only when a particular input is provided. Such code is typically deliberately hidden and can easily be made statistically invisible to black box testing approaches. For example, if the function is triggered only when a particular input sting is provided the probability of detection by black box testing is 1 in 88^n, where n is the number of characters in the input and we are assuming all the characters on a standard keyboard can be used. For a field of length 30 the number of test cases needed to detect such a path this is about 2.16×10^{58}, which is not technically or economically feasible.

However, a branch coverage criterion coupled with a constraint logic solver for finding test cases to exercise infrequent branches has be found to be effective at detecting such faults (Molnar, 2008).

Conclusions

Effective and cost-efficient testing for US Navy OA software can be achieved by a mixture of automation methods to determine which tests can be safely eliminated by reusing previous test results, and methods for choosing test cases that are most likely to expose errors without duplicating coverage of other test cases.

This paper explains how automated testing can be systematically performed based on historical data, in a way that exposes the most frequently manifesting errors earliest in the process. We also identify some of the weaknesses of purely statistical approaches to testing and identify methods for overcoming these weaknesses.

References

Bayesian information criterion. (n.d.). Retrieved March 2010, from Wikipedia, http://en.wikipedia.org/wiki/Bayesian_information_criterion

Berzins, V. & Dailey, P. (2009). How to check if it is safe not to retest a component. In *Proceedings of the Sixth Annual Acquisition Research Symposium*. Monterey, CA*:* Naval Postgraduate School.

Cramér-von-Mises criterion (n.d.). Retrieved March 2010, from Wikipedia, http://en.wikipedia.org/wiki/Cram%C3%A9r%E2%80%93von-Mises_criterion

Dailey, P., & Luqi. (2010, March). *Profile-based automated testing process for open architecture track-processing software* (NPS-CS-10-005). Monterey, CA: Naval Postgraduate School.

Dailey, P. (2010, June (est.)). *High-fidelity profile-based automated testing of open architecture track-processing software.* Unpublished doctoral dissertation. Monterey, CA: Naval Postgraduate School.

Dfittool. (n.d.). Retrieved March 2010, from MathWorks^{TM}, http://www.mathworks.com/access/helpdesk/help/toolbox/stats/dfittool.html

Kernel density estimation. (n.d.). Retrieved October 2009, from Wikipedia, http://en.wikipedia.org/wiki/Kernel_density_estimation

Molnar, D., Godefroid, P., Levin, M. (2008). Active property checking. *EMSOFT 2008.*

Musa, J.D. (1993). Operational profiles in software reliability engineering. *IEEE Software*, *10*(2), 14–32*.*

Trentin, E. (2006). Simple and effective connectionist nonparametric estimation of probability density functions. *ANNPR 2006, LNAI 4087*, 1–10.

Voas, J. (2000, March/April). Will the real operational profile please stand up? *IEEE Software*, 87–89.

2003 - 2010 Sponsored Research Topics

Acquisition Management

- Acquiring Combat Capability via Public-Private Partnerships (PPPs)
- BCA: Contractor vs. Organic Growth
- Defense Industry Consolidation
- EU-US Defense Industrial Relationships
- Knowledge Value Added (KVA) + Real Options (RO) Applied to Shipyard Planning Processes
- Managing the Services Supply Chain
- MOSA Contracting Implications
- Portfolio Optimization via KVA + RO
- Private Military Sector
- Software Requirements for OA
- Spiral Development
- Strategy for Defense Acquisition Research
- The Software, Hardware Asset Reuse Enterprise (SHARE) repository

Contract Management

- Commodity Sourcing Strategies
- Contracting Government Procurement Functions
- Contractors in 21st-century Combat Zone
- Joint Contingency Contracting
- Model for Optimizing Contingency Contracting, Planning and Execution
- Navy Contract Writing Guide
- Past Performance in Source Selection
- Strategic Contingency Contracting
- Transforming DoD Contract Closeout
- USAF Energy Savings Performance Contracts
- USAF IT Commodity Council
- USMC Contingency Contracting

Financial Management

- Acquisitions via Leasing: MPS case
- Budget Scoring
- Budgeting for Capabilities-based Planning

- Capital Budgeting for the DoD
- Energy Saving Contracts/DoD Mobile Assets
- Financing DoD Budget via PPPs
- Lessons from Private Sector Capital Budgeting for DoD Acquisition Budgeting Reform
- PPPs and Government Financing
- ROI of Information Warfare Systems
- Special Termination Liability in MDAPs
- Strategic Sourcing
- Transaction Cost Economics (TCE) to Improve Cost Estimates

Human Resources

- Indefinite Reenlistment
- Individual Augmentation
- Learning Management Systems
- Moral Conduct Waivers and First-tem Attrition
- Retention
- The Navy's Selective Reenlistment Bonus (SRB) Management System
- Tuition Assistance

Logistics Management

- Analysis of LAV Depot Maintenance
- Army LOG MOD
- ASDS Product Support Analysis
- Cold-chain Logistics
- Contractors Supporting Military Operations
- Diffusion/Variability on Vendor Performance Evaluation
- Evolutionary Acquisition
- Lean Six Sigma to Reduce Costs and Improve Readiness
- Naval Aviation Maintenance and Process Improvement (2)
- Optimizing CIWS Lifecycle Support (LCS)
- Outsourcing the Pearl Harbor MK-48 Intermediate Maintenance Activity
- Pallet Management System
- PBL (4)
- Privatization-NOSL/NAWCI
- RFID (6)

- Risk Analysis for Performance-based Logistics
- R-TOC AEGIS Microwave Power Tubes
- Sense-and-Respond Logistics Network
- Strategic Sourcing

Program Management

- Building Collaborative Capacity
- Business Process Reengineering (BPR) for LCS Mission Module Acquisition
- Collaborative IT Tools Leveraging Competence
- Contractor vs. Organic Support
- Knowledge, Responsibilities and Decision Rights in MDAPs
- KVA Applied to AEGIS and SSDS
- Managing the Service Supply Chain
- Measuring Uncertainty in Earned Value
- Organizational Modeling and Simulation
- Public-Private Partnership
- Terminating Your Own Program
- Utilizing Collaborative and Three-dimensional Imaging Technology

A complete listing and electronic copies of published research are available on our website: www.acquisitionresearch.org

THIS PAGE INTENTIONALLY LEFT BLANK

Improved Software Testing for Open Architecture

Valdis Berzins

Professor of Computer Science, Naval Postgraduate School

Agenda

- **Research Context**
  - U.S. Navy Open Architecture
  - Problem Addressed & Proposed Solution
- **Profile-Based Automated Software Testing**
  - Automated Testing Process
  - HFPM Functional Concept
  - Acquisition Community HFPM Employment
  - Deriving HFPs from Historical Data
  - Validating HFPs
  - Deriving Stress-Testing HFPs from Historical Models
- **Conclusions**

U.S. Navy Open Architecture

- **A multi-faceted strategy for developing joint interoperable systems that adapt and exploit open system design principles and architectures**

- **OA Principles, processes, and best practices:**
  - Provide more opportunities for completion and innovation
  - Rapidly field affordable, interoperable systems
  - Minimize total ownership cost
  - Maximize total system performance
  - Field systems that are easily developed and upgradable
  - Achieve component software reuse

Problem and Proposed Solution

- **Traditional U.S. Navy Software T&E practices will limit many benefits of OA**
  - It will be virtually impossible to field frequent and rapid configuration changes

- **New Testing Technologies, Processes & Policies are Needed**
  - Determine how to Safely Reduce Amount of Testing Required (Berzins, 2009)
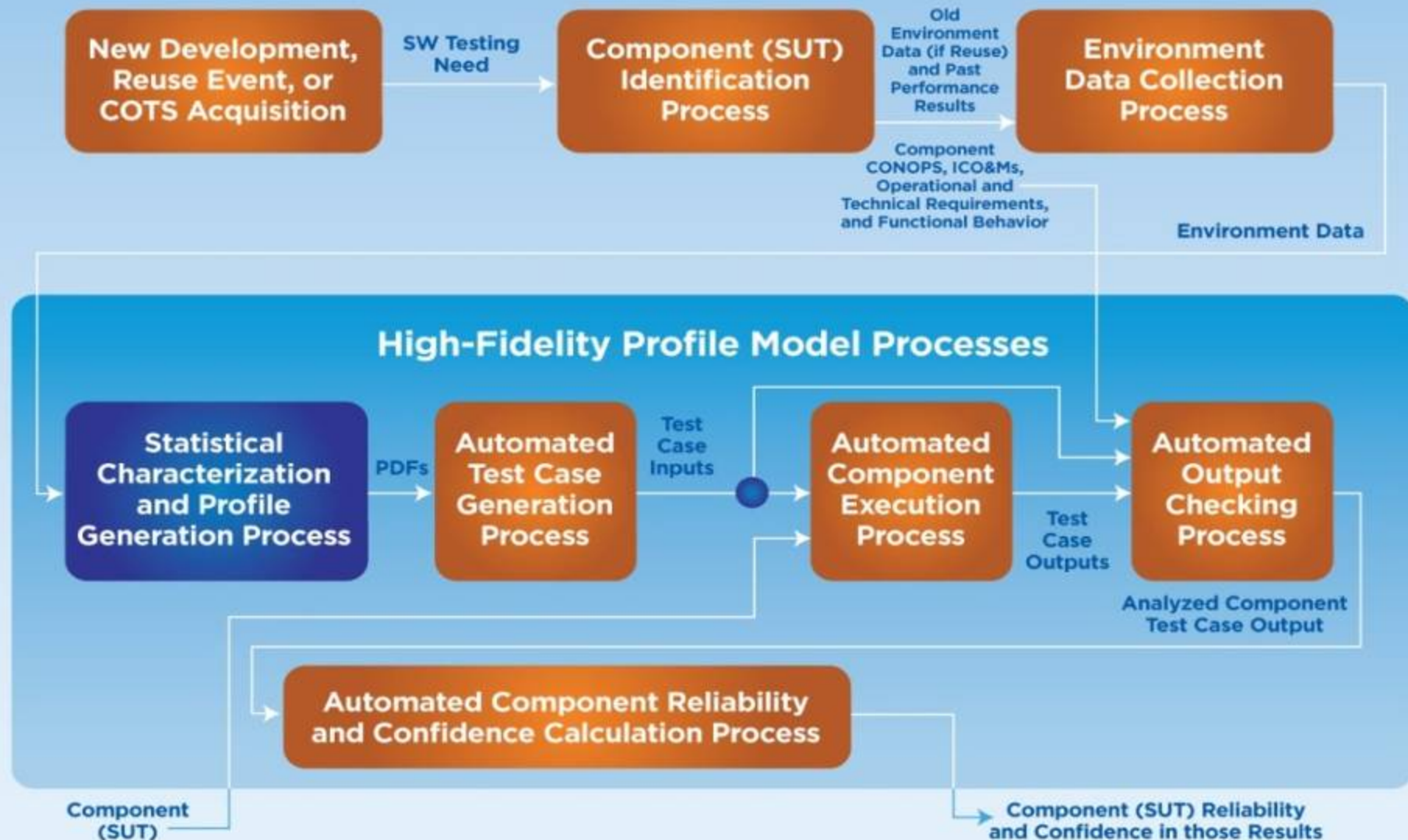 - **Transition from Manual Testing to Profile-Based Automated Statistical Testing**

HFPM-Based Automated Software Testing Process

- Software's requirements, CONOPS, architecture standards, and interfaces used to establish boundaries for component testing

- Component's external environment analyzed and characterized

- Environment statistical model (HFPM) used to automatically generate test cases, execute test cases and check component outputs

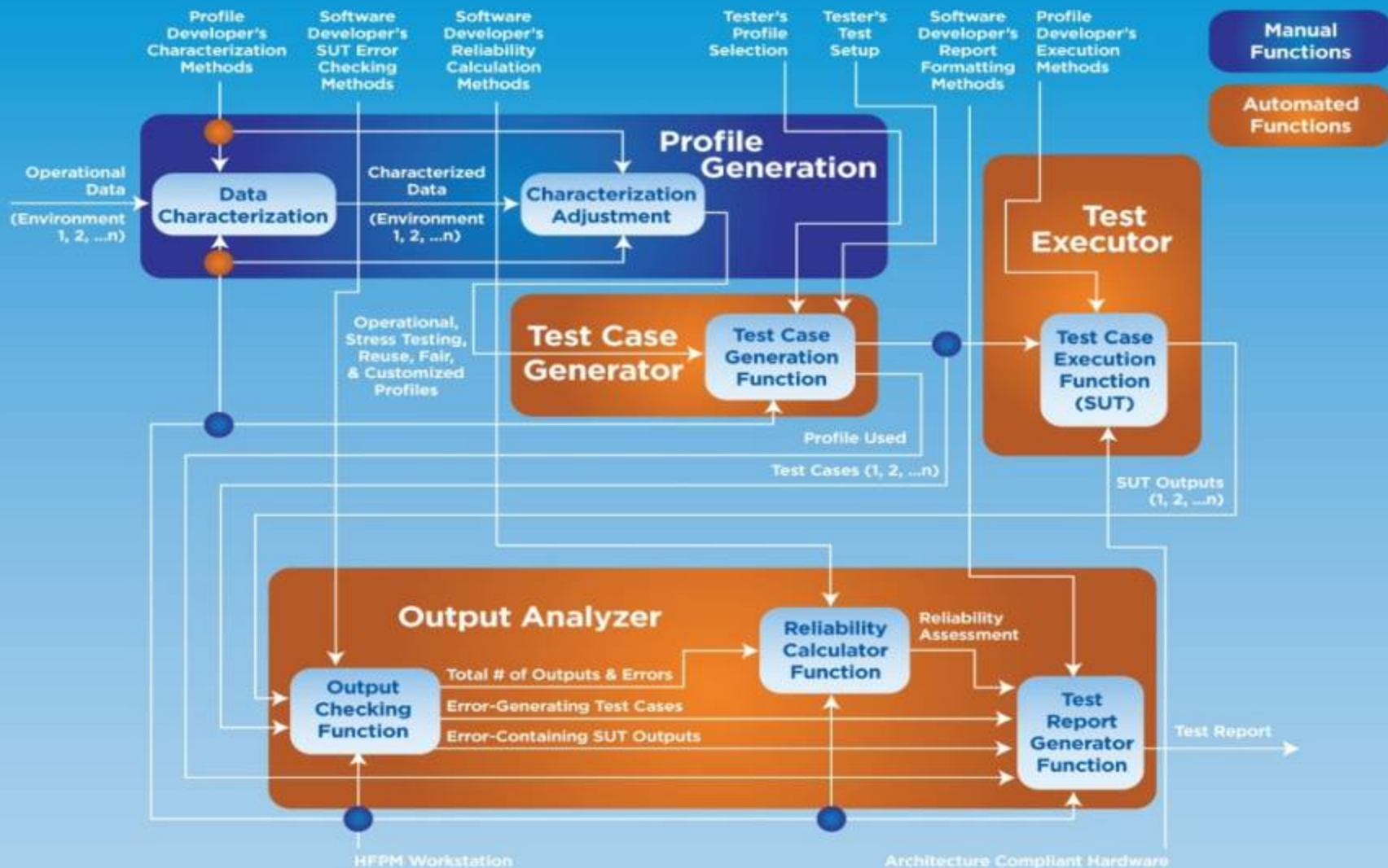Effective for new development efforts, reuse, or COTS acquisition

High-Fidelity Profile Model (HFPM)

- **HFPM utilizes statistical environment characterizations to automatically test software**
  - Profile-Based => Optimized test case coverage
  - Automated => High #s of test cases => High confidence in results
  - Concept is <u>scalable</u> from component to system level

- **Model is reusable, following component throughout life-cycle**
  - Profiles can be modified to check component behavior in multiple environments and at different stress-levels
  - Model can be used to check multiple component configurations during iterative development
  - Model architecture is reusable

- **HFPM developed to accompany each component during testing**
  - Initial investment up front enables long-term benefits including reduction in testing time and more effective & efficient testing

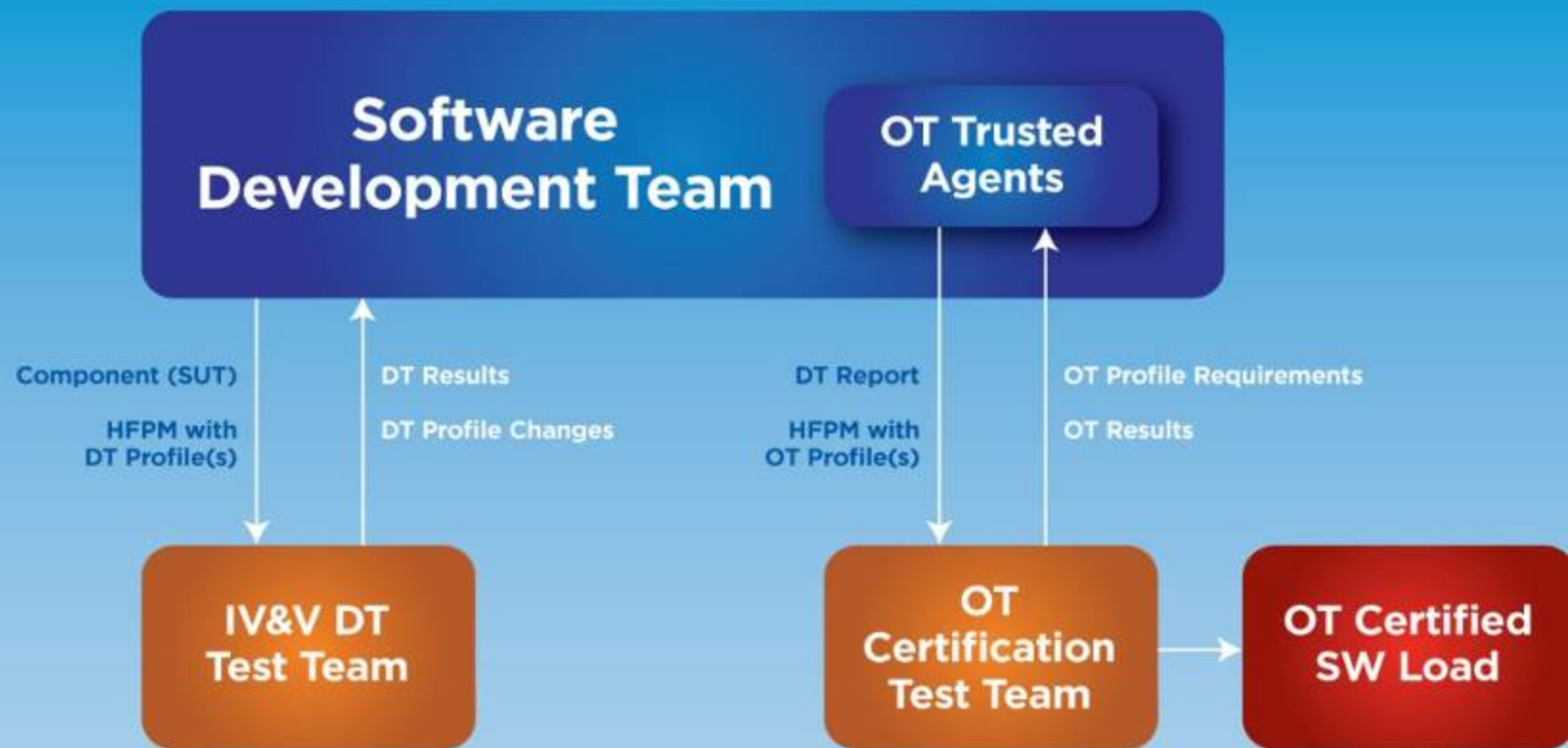# High-Fidelity Profile Model Functional Concept

HFPM-Based Testing Employment for U.S. Navy Acquisition

- **HFPM developed and used during new software development, COTS acquisition, or reuse event by R&D team**
  - R&D DT profiles include stress-testing profiles

- **Component, HFPM & profiles passed to IV&V for developmental testing (DT)**
  - IV&V team can use R&D profiles or modify if desired
  - R&D / IV&V DT loop continues until software is mature

- **Mature component, HFPM & DT report passed to certification team for operational testing (OT)**
  - Certification team defines OT requirements
  - R&D OT trusted agents develop operational profiles
  - Cert team conducts/witnesses OT and certifies component or sends back for more development & DT

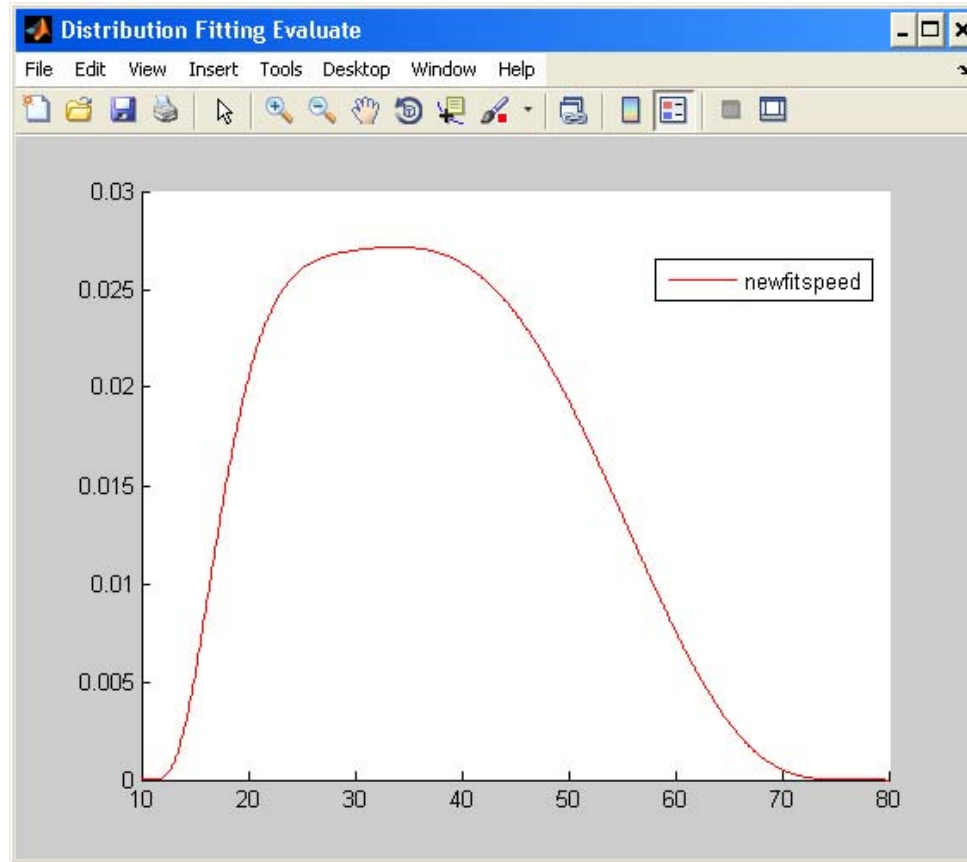HFPM-Based Automated Software Testing Process Employment Scheme

Deriving HFPs from Historical Data

- ## Collecting historical data
  - Lots of real data is best
  - Else can approximate using known constraints

- ## Characterizing historical data
  - Maximum Likelihood parameter estimation
  - Maximum *A Posteriori* probability estimation
  - Kernel density Estimation
  - Parzen Neural Network

Example: Maritime tracks

Notional Small Boat Maximum Velocity PDF (Knots) (Dailey, 2010)

Validating HFPs

- Bayesian Information Criterion
  - Minimize (K ln N – 2 ln L)
    - K: number of free parameters to be estimated
    - N: number of data points
    - L: maximum of the likelihood function for the estimated model

- Goodness of Fit Tests
  - Minimize sum of squared error

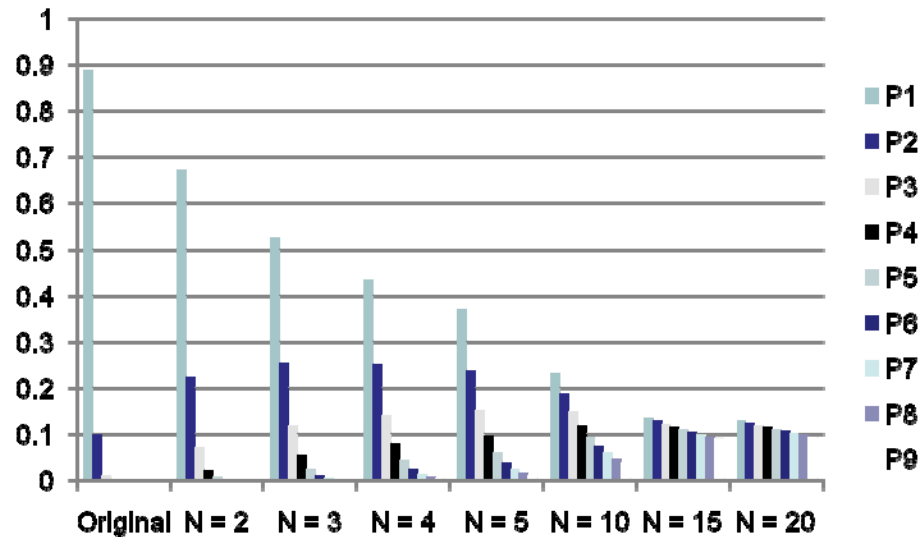- Confidence calculation based on amount of historical data

Deriving Stress-Testing HFPs from Historical Models

- Standard deviation-based methods
- Scale-expanding transformations
  - $P(x-m) \rightarrow P((x-m)/s)$, $s \in \{10, 100, 1000, \ldots\}$
  - Work for numerical and vector types
- Probability scaling transformations
  - $P(x) \rightarrow P(x)^{1/n}$, $n \in \{2, 3, \ldots, 20\}$
  - Work for arbitrary data types
- Utilization of dominating test cases
- Defining coverage criteria

Example: Probability Scaling Transformation

	Original	N = 2	N = 3	N = 4	N = 5	N = 10	N = 15	N = 20
P1	0.88888889	0.670925	0.526601	0.432891	0.369481	0.233181	0.134859	0.128692
P2	0.1	0.225035	0.254214	0.250707	0.238684	0.187417	0.128468	0.12409
P3	0.01	0.071162	0.117996	0.140983	0.150599	0.14887	0.12206	0.119418
P4	0.001	0.022504	0.054769	0.079281	0.095022	0.118252	0.115971	0.114922
P5	0.0001	0.007116	0.025421	0.044583	0.059955	0.093931	0.110186	0.110595
P6	0.00001	0.00225	0.0118	0.025071	0.037829	0.074612	0.10469	0.106432
P7	0.000001	0.000712	0.005477	0.014098	0.023868	0.059266	0.099468	0.102425
P8	0.0000001	0.000225	0.002542	0.007928	0.01506	0.047077	0.094506	0.098568
P9	0.00000001	7.12E-05	0.00118	0.004458	0.009502	0.037395	0.089792	0.094857

Dominating Cases for Stress Testing

Error Type	Heuristics for choosing stress test cases
Numeric Overflow	Largest and smallest representable numbers
Buffer Overflow	Very long input string
Free Storage Overflow	Create many new objects
Wrong Conditional Logic	Data values close to the both sides of an interval boundary
Unprotected Pointers	Null pointer
Unprotected Division	Zero

Conclusions

- **Effective and cost-efficient testing can be achieved by a mixture of automation methods**
  - Determine which tests can be safely eliminated
  - Determine which test cases will most-likely expose errors

- **This research defines a statistically-based automated testing process that can be executed using historical environment data**
  - Reduces testing time while increasing coverage
  - Model-driven process is reusable, scalable
  - Process should enable benefits brought on by OA